
# Infrastructure as data with Ansible for easier Continuous Delivery

**Carlo Bonamico**
carlo.bonamico@nispro.it
http://www.carlobonamico.com/blog/
@carlobonamico

**OSS4B 2013 - Open Source Software for Business**
19-20 September 2013, Monash University Prato Centre
Prato, Tuscany, Italy

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Do you like

- Staying up late to reconfigure a server?
- Having to rely on a server that took a week to setup,
  - and lose it because of an HD failure?
- Worrying about late and “frightening” releases?
- Spend hours/days troubleshooting
  - Only to discover a minor difference in OS/app config
- Being unable to deploy a critical fix
  - because the upgrade process is so fragile and long that “it is better not to touch the system”?
- Be unable to quickly scale your application on multiple servers
  - because the IT administration becomes too complex and time-consuming?

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible Hello World

If the answer to these question is

NO!

Then this talk is for you...

*Discuss how to improve our software delivery...*

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Or better...

- Find a better way of delivering **value** to users / customers
  - Reduce the time between when new needs arise and when IT satisfies them
  - Even anticipating them...
- *How long does it take to deploy a single line change into production?*
  - (crucial question by Mary Poppendieck)
- Any work (code, configuration, setup...) available only to developers and testers, but not to users
  - Is (almost) worthless!

We need
**a better process**
(and some tools)

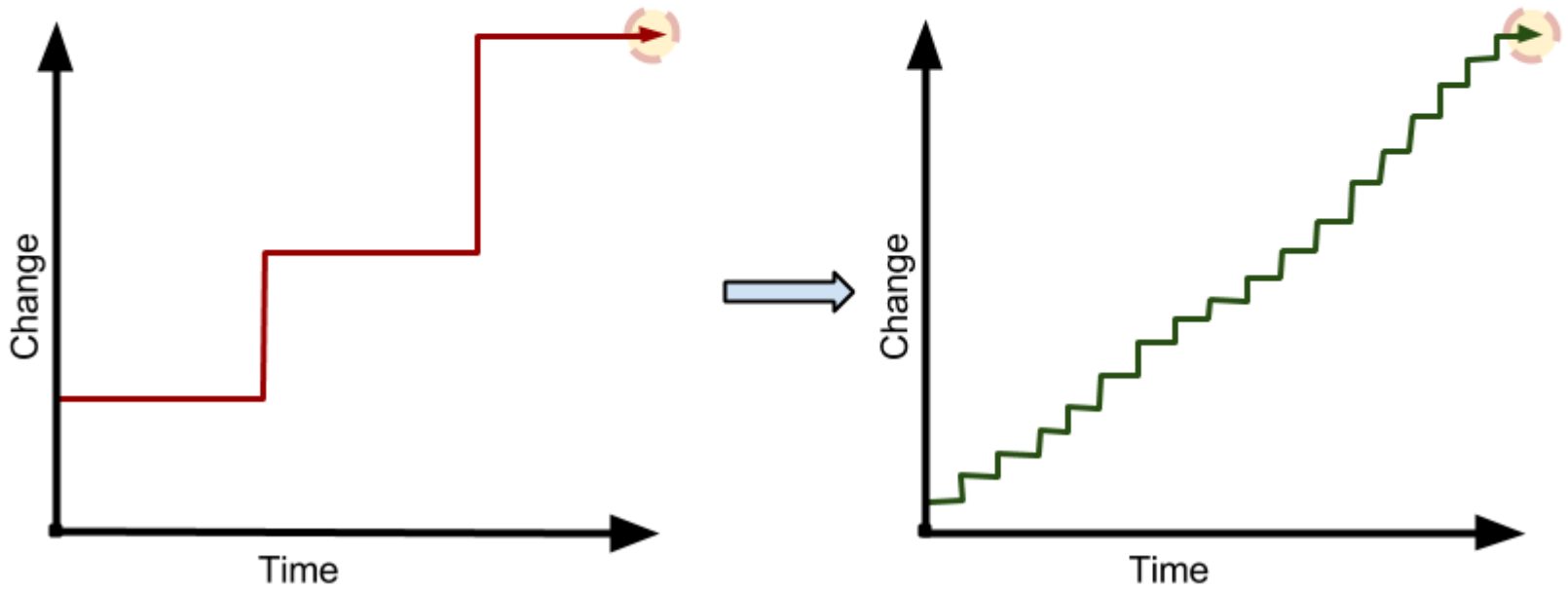
OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Why this is not easy?

- Time spent on tasks not effective
  - waste in Lean terms
- Long Test times and costs ↔ Low Quality
- Duration and complexity of deployments
  - Releasing/Deployment means Risk
  - So to avoid risk, we deploy less often...
- → This actually increases risk!

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# RELEASE RISK

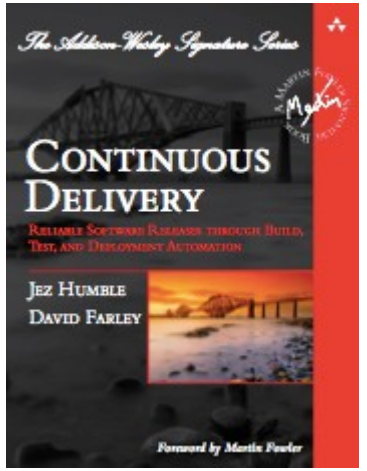


OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

Our highest priority is to satisfy the customer
through early and continuous delivery of valuable software
*The Agile Manifesto*

# Continuous Delivery

- Avoid huge manual changes
  - 1-2 macro releases per year
  - Each requiring seven people each one of which has intimate detail of a single issue in the process
- If they go wrong, long service outages
  - Unacceptable risk level
  -
- Transition to many incremental steps
  - Every week / day / even hours...
  - Less risky (and more easily rolled back)
  - Ideally automated
- Jez Humble, David Farley
- 
  - http://www.slideshare.net/jezhumble/continuous-delivery-5359386



# Advantages

- Widely adopted at
  - IBM, Amazon, Facebook, Google, ...
  - Premium Credit Limited, London Multi-Asset Exchange, Commonwealth Bank
  - RCS Media Group, Siemens, HP
- Best practice recommended by IBM and Toughtworks among many others.

*WHY?*

- IT + Business managed and improved together
  - Decrease Lead Time
  - Increase opportunities for Learning / Innovation / Optimization
- Achieve both Quality AND Productivity

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Principles of Continuous Delivery

- The process for releasing/deploying software MUST be repeatable and reliable
- Automate everything!
- If somethings difficult or painful, do it more often
- Keep everything in source control
- Done means “released”
- Build quality in
- Everybody has responsibility for the release process
- Improve continuously

- http://java.dzone.com/articles/8-principles-continuous

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Continuous Delivery Practices

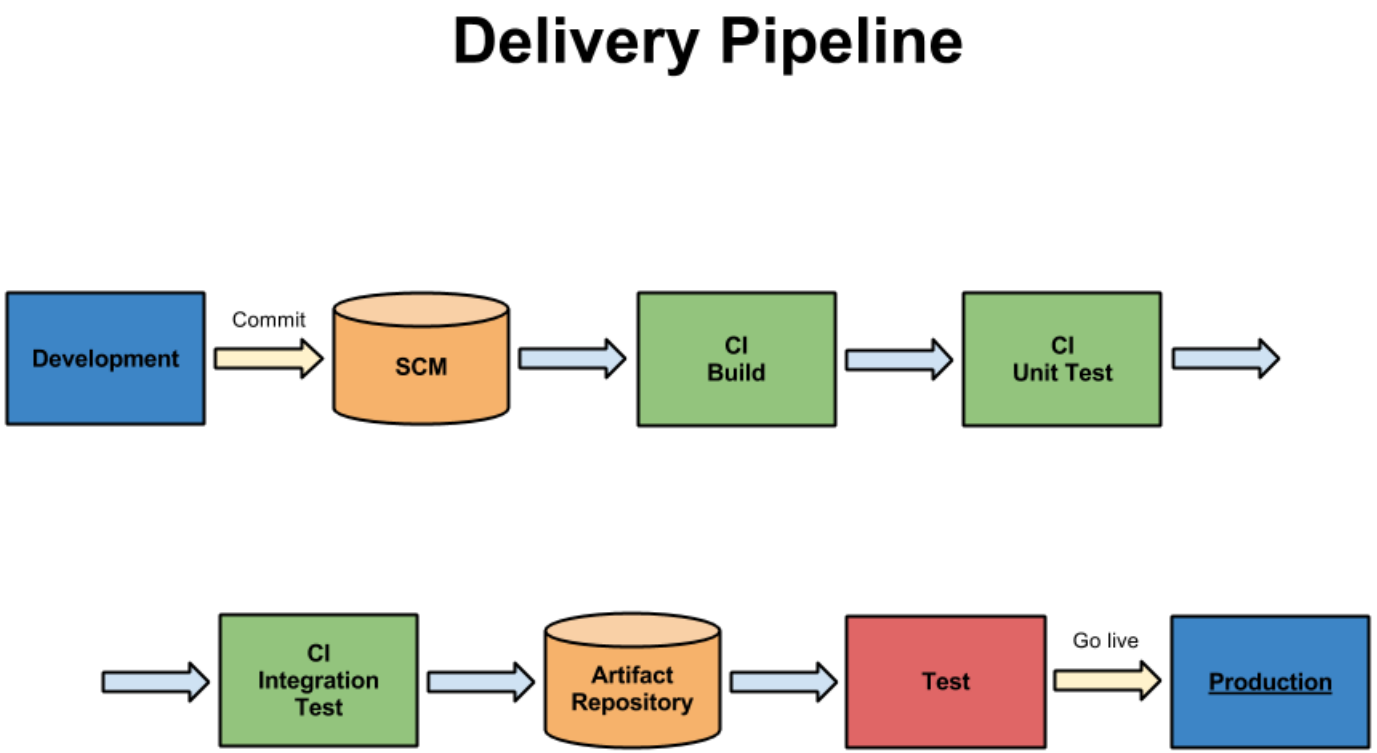
- Build binaries only once
- Use precisely the same mechanism to deploy to every environment
- Smoke test your deployment
- If anything fails, stop the line!
- Low Risk Releases
  - incremental
  - decouple deployment and release
  - focus on reducing batch size
  - optimize for resilience
- See also
  - http://continuousdelivery.com/2012/02/four-principles-of-low-risk-software-releases/

# Ok, I want to do Continuous Delivery...

  - to get advantages in term of efficiency, quality and reduction of time-to-market
- The part about incremental code changes and releases is familiar to me
  - See Continuous Integration, Agile methods
- But how do I Continuously evolve my infrastructure?
  - OS configuration
  - Installed packages
  - Security settings
  - Performance tuning
  - …
- This still requires manual work by ops people...
  - We do not have time/resources to change our way of working

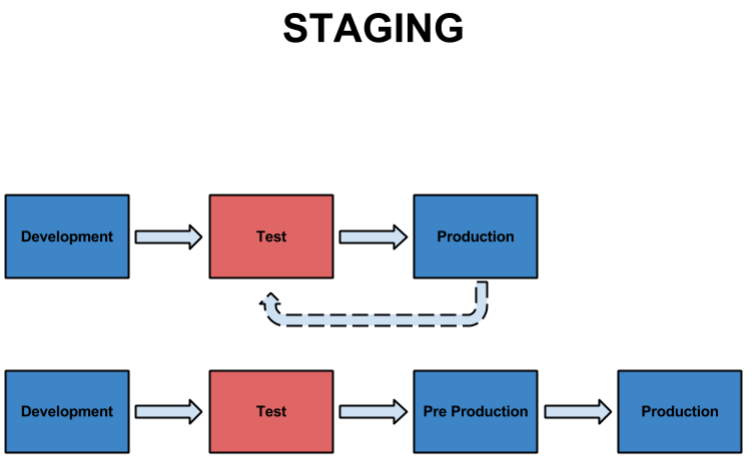
# What would I need...

- Manage BOTH code and infrastructure through a



# The ideal Infrastructure management process is

- Automatic
- Repeatable and consistent
  - Recreate as many time as needed
    - Across environments (DEV → TEST → PROD)
    - Across hosts (clusters - cloud)
- Versionable
- Robust and resilient to
  - network failures
  - hardware failures
- Self-checking



OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

Simplicity – the art of maximizing the amount of work NOT done – is essential
*The Agile Manifesto*

# AND "easy"

- In addition to that, lightweight, EASY to setup, use and learn
  - with limited or no additional effort with respect to manual approaches
  - by both Developers and Operations
  - lending itself to incremental introduction
  -
- What if infrastructure management was easier than hacking with keyboard and shell scripts?

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible

- What if setting up and configuring your cloud / private infrastructure were even simpler than writing a shell script?
- Based on the concept of Infrastructure as Data
  - simplicity as key design requirement
  - powerful
  - easy to learn for Dev and Ops people alike
- Created by Michael De Haan of Cobbler fame
  - Open Source @ https://github.com/ansible/ansible/
  - now supported by http://www.ansibleworks.com/
- Well documented
- Growing, active and supportive community

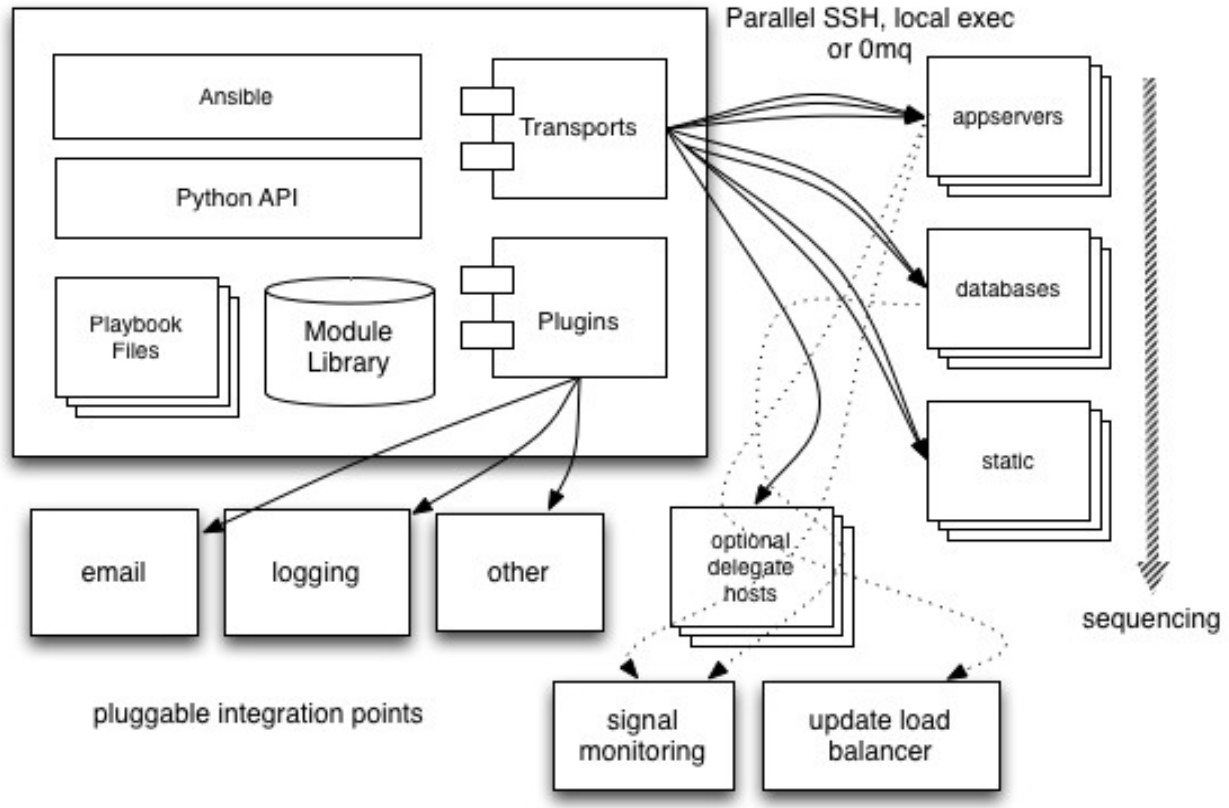
OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible provides

- Node inventory
- Three key capabilities:
  - remote execution across multiple machines
    - actions and commands
  - file, package and configuration distribution
  - automated configuration and deployment orchestrated across machines

# What's inside?

- Modular, agentless, secure architecture
  - Python core, modules in any language



OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# How does Ansible work?

- Work on all Unix/Linuxes
  - currently limited Windows support
- Transport over SSH by default
  - or encrypted socket for performance
    - *accelerated mode*
  - pluggable transport protocols
- Text-based and versioning friendly
  - inventory, configuration and playbooks in YAML
- No DB is involved
  - but CMDBs can be integrated via API

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Getting Ansible

- Minimal install
  - `sudo add-apt-repository ppa:rquillo/ansible`
  - `sudo apt-get update`
  - `sudo apt-get install ansible -y`
- Or even from git checkout or pip
  - http://www.ansibleworks.com/docs/gettingstarted.html
- Minimal requirements
  - Python 2.6 on the commander
  - Python 2.4 on the nodes
  - Three phyton packages (autoinstalled by pip or apt )
    - Paramiko, Jinjia2, PyYaml

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Pizzamatic Time!

- Infrastructure for a fictional pizza-order-taking SAAS

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Pizzamatic infrastructure

- We start from “empty” Ubuntu Server images and add
  - front-end server with Apache2 and mod_proxy
  - back-end application servers with Tomcat 7
  - Postgresql DB
- Common features
  - Ssh public key – passwordless login
  - Ufw for firewall
- Assume initial user/pwd: manager/ansibledemo

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Getting Started

- Configure /etc/hosts or DNS
- Configure ansible_hosts
  - .ini format
  - Hosts
  - Groups, with []
- Global in ~/ansible_hosts
- Local with -i <<path to ansible_hosts>>

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# First steps

- `ansible -k -m ping -u manager pizzamatic-fe-test-01`
  - -k means ask password
  - -m means module (ping)
  - -u initial connection user
  - Target host

# First steps (with Public Key)

- Setup passwordless initial login
  - `ssh-keygen -b 2048`
    - » enter pizzamatic_rsa as filename
  - `ssh-add ~/.ssh/pizzamatic_rsa`
  - `ssh-copy-id -i ~/.ssh/pizzamatic_rsa manager@pizzamatic-ge-test-01`
- Then
  - `ansible -m ping -u manager pizzamatic-fe-test-01`
- If it hangs, either
  - You forgot the -k, and a certificate was not installed (or viceversa)
  - You added the -K (sudo password), and passwordless sudo is enabled

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Move to Playbooks

- Efficient way of describing the desired configuration of multiple hosts
  - And then “apply” it
  - Incrementally
    - Auto-resume
    - Synchronization
    - Versioning
  - Composition with *Roles*
- ansible-playbook pizzamatic.playbook

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Behaviour Driven Development - with Infrastructure???

- First, descrive desired infrastructure status as plain text
  - `#pizzamatic service requires front-end`
    - `#pizzamatic service requires apache2`
    - `#etc`
  - `#pizzamatic service requires application servers`
- Then translate it incrementally in ansible “actions” → execute it!

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Actions: an example

```
#Installing and configuring Apache 2
  - name: require Apache2 package installed
    action: apt pkg=apache2

  - name: Generate the virtual host configuration
    action: template src=src/${service.name}-ssl.j2
dest=/etc/apache2/sites-available

  - name: Ensure the site is up (custom command)
    action: command a2ensite ${service.name}-ssl

  - action: service name=apache2 state=started
```

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible Actions

- Not ideal term! Very often “actions” do nothing!
  - Because the system is already in the desired state
    - `action: file dest=/home state=present`
- They do something only if the system is not in the desired state

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible Actions

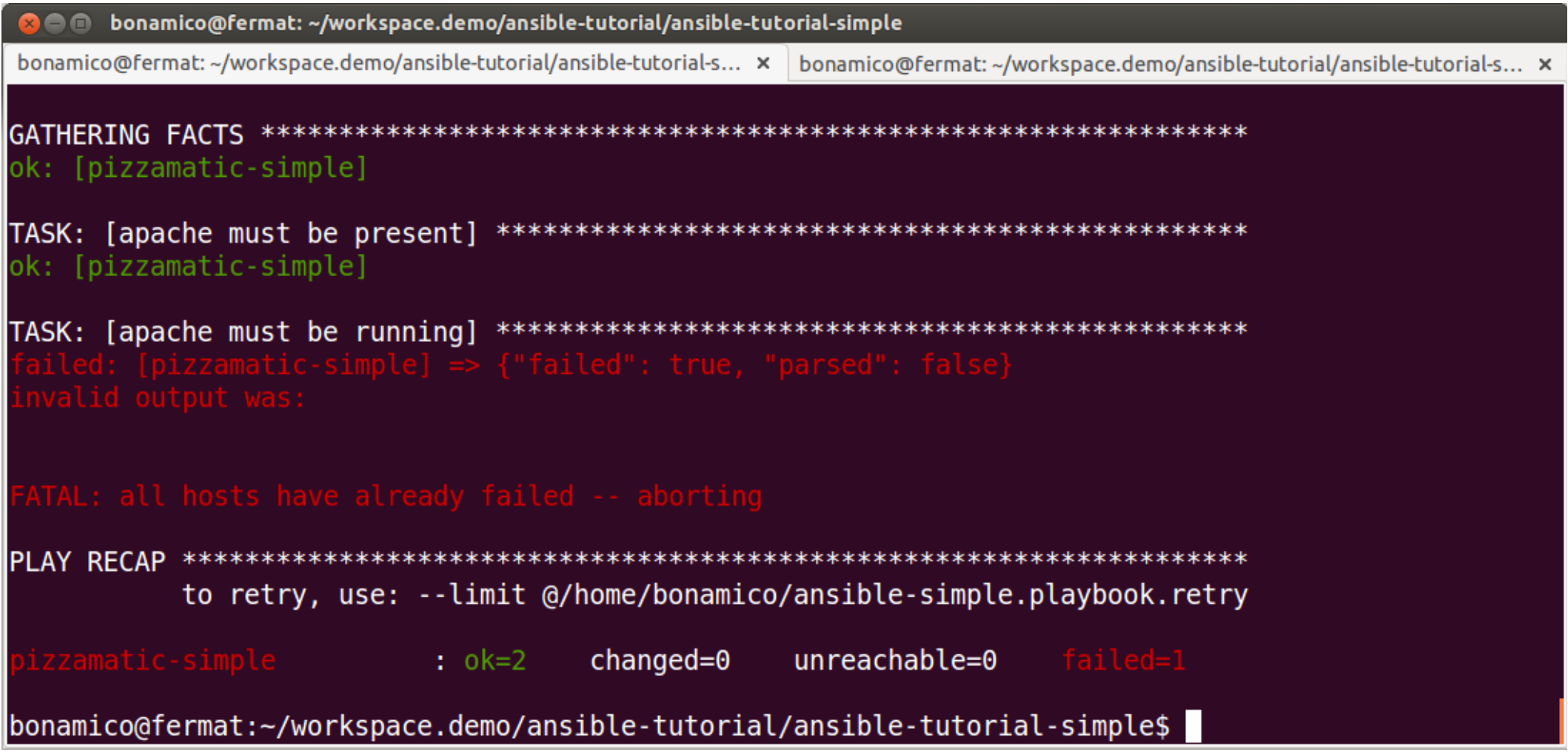
- Most Ansible Actions are Idempotent
  - “big word” meaning that you can repeat them as many times as you want and always get the same result
- In the real world
  - Turning a glass upside down to empty it is idempotent
  - Filling a glass with water is NOT idempotent
- In practice, it's what makes ansible useful

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# BDD with Infrastructure???

- Red
  - Error
- Yellow
  - Applied, changed
- Green
  - Already in the desired state

# Red: error


bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-simple
bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-s...
bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-s...
GATHERING FACTS **********
ok: [pizzamatic-simple]
TASK: [apache must be present] **********
ok: [pizzamatic-simple]
TASK: [apache must be running] **********
failed: [pizzamatic-simple] => {"failed": true, "parsed": false}
invalid output was:
FATAL: all hosts have already failed -- aborting
PLAY RECAP **********
to retry, use: --limit @/home/bonamico/ansible-simple.playbook.retry
pizzamatic-simple : ok=2 changed=0 unreachable=0 failed=1
bonamico@fermat:~/workspace.demo/ansible-tutorial/ansible-tutorial-simple$

# Yellow: change performed

bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-simple

bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-s... × | manager@ubuntu: ~ ×

```
bonamico@fermat:~/workspace.demo/ansible-tutorial/ansible-tutorial-simple$ ansible-playbook -k -K -
i ../ansible_hosts ansible-simple.playbook
SSH password:
sudo password [defaults to SSH password]:

PLAY [pizzamatic-simple] ******************************************************

GATHERING FACTS ***************************************************************
ok: [pizzamatic-simple]

TASK: [apache must be present] ************************************************
ok: [pizzamatic-simple]

TASK: [apache must be running] ************************************************
changed: [pizzamatic-simple]

PLAY RECAP ********************************************************************
pizzamatic-simple          : ok=3    changed=1    unreachable=0    failed=0

bonamico@fermat:~/workspace.demo/ansible-tutorial/ansible-tutorial-simple$
```

# Green: state already ok

bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-simple

bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-s... × | bonamico@fermat: ~/workspace.demo/ansible-tutorial/ansible-tutorial-s... ×

```
bonamico@fermat:~/workspace.demo/ansible-tutorial/ansible-tutorial-simple$ ansible-playbook -k -K -
i ../ansible_hosts ansible-simple.playbook
SSH password:
sudo password [defaults to SSH password]:

PLAY [pizzamatic-simple] ******************************************************

GATHERING FACTS ***************************************************************
ok: [pizzamatic-simple]

TASK: [apache must be present] ************************************************
ok: [pizzamatic-simple]

TASK: [apache must be running] ************************************************
ok: [pizzamatic-simple]

PLAY RECAP ********************************************************************
pizzamatic-simple          : ok=3    changed=0    unreachable=0    failed=0

bonamico@fermat:~/workspace.demo/ansible-tutorial/ansible-tutorial-simple$
```

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Infrastructure as what?

Ansible = Infrastructure as Data

You describe your infrastructure

You version the description

“Applying” the description and actually ensuring that the infrastructure exists and is in the desired state is an implementation detail

- (and up to ansible, not you)

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

# Ansible Modules

- Clean and modular way of defining actions
  - encapsulate best practices
  - A single ansible action encapsulates lines and lines of shell scripts
- Very strong emphasis on reuse
  - And abstraction

OSS4B (OPEN SOURCE SOFTWARE FOR BUSINESS)

And many more!

# Ansible Modules

- Implemented in any language
  - Python, java, bash...
  - Core modules are in python
- Input:
  - parameter string
- Output:
  - json data
  - Including state

- Cloud
  - ec2, linode, openstack, rackspace, vmware
- Commands
  - command, shell, ...
- Database
  - mongodb, mysql, postgresql, redis...
- Files
  - assemble, copy, fetch, lineinfile, template
- Inventory
  - add_host, group_by

# Ansible Modules

- Messaging
  - rabbitmq
- Monitoring
  - airbrake, boundary, datadog, nagios, pagerduty...
- Net Infrastructure
  - arista, dnsmadeeasy, ...
- Network
  - get_url, slurp, uri (REST client)
- Notification
  - mail, irc, jabber,...
- Packaging
  - apt, easy_install, gem, macports, npm, openbsd, pip, rpm, yum, ...
- Source Control
  - bzr, git, hg, subversion
- System
  - authorized_key, cron, facter, filesystem, group, lvg/lvol, mount, selinux, service, user
- Utilities
  - accelerate, debug, wait_for
- Web
  - django_manage, htpasswd, supervisorctl

# Variables

- Declared
  - In the ansible_hosts file
  - individual YAML files relative to the inventory file
    - e.g. host_vars/pizzamatic-fe-test-01

```
---
ntp_server: acme.example.org
```

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Facts

- Automatically collected facts about systems involved in the playbook
  - `${inventory_hostname}`
  - `${ansible_eth0.ipv4.address}`
- Can be use as variables in playbook and templates
- Check with `ansible <<hosts>> -m setup`
- Can extended with commander fact database (in facts.d)

# Templates

- Jinja2 templates
  - very similar to java ${property} syntax
- Env.sh.j2

```
export JAVA_HOME=/home/
{{service.user}}/jdk1.7.0
export PATH=$PATH:$JAVA_HOME/bin
```

# Handlers

- Respond to asynchronous events

```
- template: src=template.j2
dest=/etc/foo.conf
    notify:
        - restart apache

 handlers:
- name: restart apache
    action: service name=apache2
state=restarted
```

# Playbook Structure

```
---
- hosts: pizzamatic-fe-test-01
gather_facts: yes
user: pizzamatic
sudo: yes

vars_files:
  - pizzamatic.yml

vars:
  name: pizzamatic
tasks:
- include: pizzamatic-fe.playbook #child sees parent
variables and params
```

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Roles

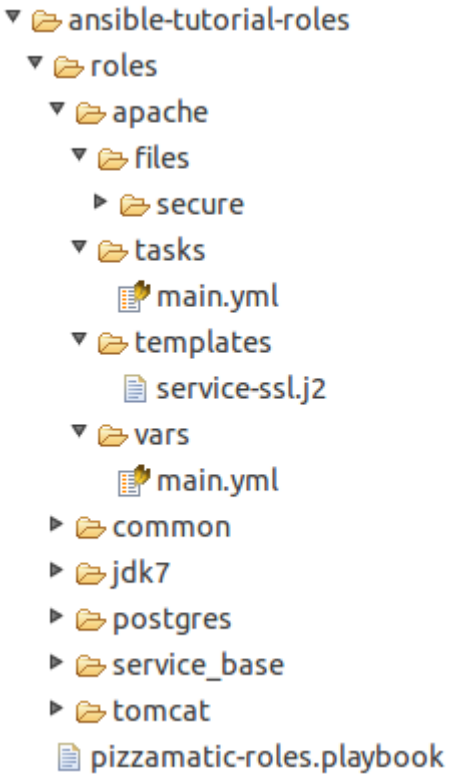
- Encapsulate a reusable service definition including
  - Actions
  - Variables
  - Templates
  - Files
- Defined in standard directory structure (subproject-like)
- A host / group can have multiple roles
- A role can be applied to multiple groups
- E.g.
  - Roles: common, postgres, tomcat, apache
  - In DEV: all roles on a single machine
  - In TEST/PROD: common on all machines, other roles to multiple machines in the cluster

# Roles example

```
---
- hosts: pizzamatic
  gather_facts: no
  sudo: yes
  user: manager

  roles:
    - common
    - jdk7
    - postgres
    - tomcat
    - apache

  tasks:
   - debug: "Generic task for {{service.user}}"
```



# File management and transfer

- To the nodes
  - `ansible atlanta -m copy -a "src=/etc/hosts dest=/tmp/hosts"`
  - `ansible webservers -m file -a "dest=/srv/foo/b.txt mode=600 owner=mdehaan group=mdehaan"`
  - `ansible webservers -m file -a "dest=/path/to/c mode=644 owner=mdehaan group=mdehaan state=directory"`
  - `ansible webservers -m file -a "dest=/path/to/c state=absent"`
- From the nodes
  - Use the fetch module

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Case Studies

# Case A) Advertising project

- Platform for deliverying innovative advertising clips to digital cinemas across Italy
  - Central server
  - N local players at each venue
- Challenge: high cost of in-site intervention
- Everything installed with Ansible
  - Including video drivers, audio volume settings…
- Automatic configuration and software upgrades
  - Even with intermittent connectivity
- Lessons learned
  - Essential for an innovative project where cycle time and reliability where needed to meet demanding business deadlines
  - Mitigated hardware problems

# Case B) Alfresco platform setup

- Used to be fairly time consuming
  - OS / LVM for storage
  - Postgresql DB
    - And tablespace configuration
  - JDK
  - Alfresco
    - Configuration
    - Active Directory integration
- Ansible script developed in 1-2 days
  - Setup time: 15 minutes
- Easily adapted to three different projects / customers
  - Even more reuse with the “roles” feature added in 1.2

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Advantages

- Significantly more reuse with respect to shell scripts
  - Comparable effort on the first install
  - Huge time saving on the following
- Tests are much more reliable
  - Exactly replicate PROD environment
- Support for incremental, always-on Green/Blue deployments

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Best Practices

- Good old Software Engineering Principles still apply!
  - *Don't Repeat Yourself*
  - Good Names make the difference
  - Be simple
  - S.O.L.I.D.
    - http://butunclebob.com/ArticleS.UncleBob.PrinciplesOfOod

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Useful Tools

- Yaml Editor for Eclipse
  - https://code.google.com/p/yedit/
  - https://code.google.com/p/yamleditor/
- Git & Mercurial

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# References - Ansible

And
the very active
google group
ansible-project

- Ansible Home & Ansible Docs
  - http://www.ansibleworks.com/docs/
- Presentations
  - https://speakerdeck.com/mpdehaan/ansible
- AnsibleWorks
  - http://www.ansibleworks.com/
- This tutorial
  - https://github.com/carlobonamico/ansible-tutorial

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# Tips and Tricks

- Speed up SSH with Control Persist
  - http://blogs.perl.org/users/smylers/2011/08/ssh-productivity-tips.html

# References - Continuous Delivery

- General
  - http://continuousdelivery.com/
  - http://www.slideshare.net/eduardsi/continuous-delivery-18191261
  - http://www.thoughtworks.com/sites/www.thoughtworks.com/files/files/us-format-continuous-delivery-brochure-online.pdf
- Maturity model
  - http://info.thoughtworks.com/rs/thoughtworks2/images/Continuous%20Delivery%20_%20A%20Maturity%20Assessment%20ModelFINAL.pdf
- Patterns
  - http://refcardz.dzone.com/refcardz/continuous-delivery-patterns

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

Thank you

for your attention!

# References

- My blog
  - http://www.carlobonamico.com
  - http://slideshare.net/carlo.bonamico
- Contact me
  - carlo.bonamico@nispro.it
- JUG Genova
  - http://juggenova.net
- Twitter
  - @carlobonamico - #ansible

OSS4B
(OPEN SOURCE SOFTWARE FOR BUSINESS)

# License

- Creative Commons:
  - Attribution-Non-Commercial-Share Alike 3.0
  - You are free:
    - to copy, distribute, display, and perform the work
    - to make derivative works
  - Under the following conditions:
    - Attribution. You must give the original author credit.
    - Non-Commercial. You may not use this work for commercial purposes.
    - Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under a licence identical to this one.